Image Compression using Morphological Representation of Wavelet Data: Review of a Paper by Servetto, Ramchandran, and Orchard

Zvika Ben-Haim

ID 034986505

February 10, 2004

Contents

1 Introduction

A picture is worth a thousand words, says the proverb. Indeed, pictures are rich sources of information: a single photograph consists of several million pixels and requires several megabytes of storage space. When designing large databases of images, such as medical records, the hardware requirements for storage and transmission of these images quickly become a major obstacle.

The objective of image compression algorithms is to reduce the storage requirements per image, while maintaining image quality (insofar as possible). Ideally, we would like to maintain the "subjective" quality of the image, i.e., the appearance of the image to a human observer. Yet by definition, the subjective quality of an image is difficult to quantify. In practice, measures such as the mean-square error (MSE) or the related peak signal-to-noise ratio (PSNR) are often used, although it is well-known that they do not always capture the subjective feeling of high-fidelity reconstruction.

While no compression algorithm can reduce file size without damage to image quality, many algorithms can reach compression ratios on the order of 25:1 (compared to an uncoded image), with barely noticeable effect on the image. Two factors contribute to this high compression ratio. First, uncoded images inherently contain much redundancy; for instance, often large sections of an image are smooth, containing nearly identical pixel values. Second, many details of an image can be changed or removed without bothering a human observer. Identifying and exploiting these psycho-physical characteristics is the main challenge in designing an image compression algorithm. The current basic strategy for image compression is described in Section 2.

Current image compression techniques differ primarily in their use of these properties to conserve bandwidth. For instance, the well-known Embedded Zerotree Wavelet (EZW) compression algorithm of Shapiro [8] is based on identifying image regions containing low coefficient values for all subbands, and coding these using a special symbol. This compression technique is reviewed in Section 2.2. Taking a different approach, Morphological Representation of Wavelet Data (MRWD) of Servetto et al. [6], the primary topic of this report, codes each subband separately, but is more flexible in its ability to code low-coefficient regions of different shapes. Two different MRWD algorithms are described in Section 3.

Section 4 contains a personal impression of the paper on morphological compression, including

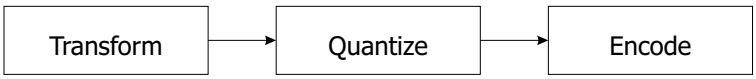

Figure 1: General image compression technique

some comments on the organization of the paper. It also details the results of various simulations performed to test the performance of the MRWD algorithms, and to compare them to standard algorithms. Finally, Section 5 summarizes this report and contains some conclusions about wavelet compression techniques, as well as suggestions for further study.

2 Image Compression Methodology

Standard source coding techniques, such as Lempel-Ziv compression [14] and arithmetic coding [13], generally perform very poorly when applied directly to an image. This is primarily because such coders assume that the symbols to be encoded are statistically independent, while this is not the case for pixels in an image: for instance, spatially close pixels have a high probability of having similar values.

Instead of directly encoding the image pixels, image compression techniques usually consist of three phases (Figure 1). First, a transform is applied to the image, in an attempt to distinguish between visually important information and unimportant information. The transform is also intended to reduce the statistical dependence between coefficients, so that source coding will be more efficient. Next, quantization is applied to the transformed coefficients. The purpose of this lossy stage is to remove or degrade those parts of the image information deemed unimportant. Finally, a standard entropy encoder is applied to the quantized coefficients.

As stated previously, an important aspect of image compression is the exploitation of characteristics of the human visual system (HVS) and of typical images, in order to remove unnecessary information from the image. One such characteristic is that most regions in an image tend to have little energy in high frequency, and this energy is usually related to such properties as texture and irregularities — properties which may be ignored by a casual observer. On the other hand, isolated high-frequency regions often represent edges, and are an important aspect of the image.

It stands to reason, then, that a successful image transform should differentiate between high-

Figure 2: Example of an image (left) and its 5-level wavelet transform (right)

frequency and low-frequency phenomena, and describe these phenomena separately for different spatial regions of the image. This requirement brings to mind the wavelet transform, which is designed specifically for describing spatially-localized frequency phenomena. Indeed, even the JPEG compression standard [11], which was designed before the use of wavelets became commonplace, uses an intuitively similar approach, in which a frequency transform (specifically, the DCT) is applied to spatially distinct blocks, in order to capture the frequency characteristics of different spatial regions in the image. The wavelet transform, however, has the additional advantage that low-frequency coefficients characterize larger regions of the image, in accordance with the fact that low frequencies describe slowly changing (and hence large) regions of the image.

Early attempts at wavelet image compression [2, 1] simply performed a wavelet transform of the data, quantized the result to obtain lossy coding, and entropy-coded the quantized coefficients. A diadic transform is usually used for this purpose, primarily for reasons of computational efficiency. After performing a wavelet transform, most high-frequency coefficients contained low-energy values which were quantized to zero, allowing them to be coded very efficiently. At the same time, meaningful high-frequency coefficients, such as those related to edges and contours, were described by high-energy coefficients and were not removed. This is demonstrated in Figure 2, in which an original image is presented along with its 5-level wavelet decomposition.

We have seen that the wavelet transform successfully describes the information content of an

image in a form which allows us to remove some types of unimportant information. However, as mentioned previously, a good image transform must also reduce the statistical dependency between the resulting coefficients. Statistical dependence between coefficients is ignored by the entropy coder, so the existence of statistical dependence indicates that better results may be achieved by alternate means. For instance, if two coefficients are known to be statistically dependent, then coding the two coefficients as a single symbol will produce a lower bitrate than coding each coefficient separately.

Observing the wavelet transform in Figure 2, it is clear that the transformed coefficients are *not* statistically independent. For example, large wavelet coefficients are clustered in complex regions of the image. Thus, the probability for a large coefficient is higher if it is surrounded by other large coefficients. The main goal of recent research [6, 8, 5] on the subject of wavelet image compression is to describe wavelet coefficients in such a way as to take advantage of this statistical dependence.

2.1 Embedded Coding

Image compression applications are often motivated by external requirements specifying that an image not exceed a certain compression ratio. For example, in an image transmission system, if the bandwidth available and desired response time are known, a requirement on the compressed file size is obtained. In other systems, the exact bandwidth may vary with network load, and may not be known in advance. The technique of embedded coding allows optimal handling of both scenarios.

Simply put, an embedded compression algorithm is one for which the transmission of bits can be stopped at any point, and the image reconstructed only from those bits which have been transmitted so far. The embedded compression algorithm is designed so that no matter where transmission is stopped, the reconstructed image is "optimal" among all other transmissions of equal length.

As has already been noted, quality (and hence optimality) of compressed images is difficult to define. The MSE is often used as a measure of quality, not because it matches the subjective feeling of high fidelity, but rather because it is analytically tractable. Given an image $\mathbf{p}$ and a reconstruction $\hat{\mathbf{p}}$, the MSE is given by

$$d = \frac{1}{N} \sum_i \sum_j (p_{ij} - \hat{p}_{ij})^2 , \tag{1}$$

where N is the number of pixels in the image. If the wavelet transform being used is unitary

(i.e., the wavelet filters are orthonormal), then the MSE is also equal to the mean squared error of the wavelet coefficients. It is therefore common to use unitary wavelet transforms for image compression. Denoting by $\mathbf{c}$ the transformed coefficients, and by $\hat{\mathbf{c}}$ the reconstructed coefficients, we have

$$d = \frac{1}{N} \sum_i \sum_j (c_{ij} - \hat{c}_{ij})^2 \,. \tag{2}$$

From this expression it is clear that transmitting the exact value of a coefficient c_{ij} decreases the MSE by c_{ij}^2. Thus, in an embedded coding scheme, larger coefficients should be transmitted first. However, the transmission of a coefficient requires trnasmitting several bits, which typically define the coefficient in an increasing resolution level. Clearly, less-significant bits contribute less to MSE reduction than do more-significant bits. Furthermore, the least-significant bit of any coefficient reduces the MSE by exactly the same amount. Hence, the optimal order in which bits are to be transmitted is the so-called *bit-plane order,* in which the most significant bit of all coefficients is transmitted first, followed by the second-most significant bit of all coefficients, and so on. In such a coding scheme, transmission can be stopped at any point, and the bits received up to that point are those bits which contribute most to MSE reduction.

To avoid problems of representations for negative coefficients, typically only the absolute value of each coefficient is coded. A special "sign symbol" is introduced together with the first nonzero bit of every coefficient, to indicate whether that coefficient is positive or negative.

It is sometimes convenient to view bit-plane encoding using the concept of *significance maps.* A significance map of a particular bit-plane is the set of all coefficients which have a value of '1' in the bit-plane, but which have a value of '0' for all bit-planes of higher significance. Formally, let us number of the bit-planes from 0 (least-significant) to M (most-significant). Then a coefficient c_{ij} is a member of the significance map of bit-plane n if, and only if,

$$2^n \leq |c_{ij}| < 2^{n+1}. \tag{3}$$

The significance map is a useful concept because it distinguishes between two statistically distinct types of bits in the bit-plane: novelty bits and refinement bits[1]. In a particular bit-plane, the *refinement bits* are those bits which were identified as significant in a previous (higher-significance)

[1]In the original paper [8], these are termed *dominant* and *subordinate* bits, respectively. I find this nomenclature misleading.

bit-plane. These bits are used to refine the values of coefficients which are already known to be nonzero. Hence one may expect their distribution to be fairly uniform. By contrast, *novelty bits* are bits which have not been described as significant up to the current bit-plane. The novelty bits are the bits at which compression is aimed, for they contain statistical properties which can be used to increase their transmission rate. Embedded compression algorithms differ primarily by the methods by which they exploit these properties.

2.2 Zerotree Coding

A naive implementation of an embedded encoder would be to compute a wavelet transform of the image, quantize this transform, and encode the quantized transform using an entropy coder, in bit-plane order. However, as mentioned in Section 2, a major performance increase can be obtained by utilitizing the statistical dependence between coefficients of the wavelet transform. The first algorithm to make use of this dependence was the Embedded Zerotree Wavelet (EZW) compression algorithm due to Shapiro [8] (see also [10]).

EZW makes use of what are known as intra-band dependencies, i.e., the statistical dependence between coefficients in one subband and coefficients describing the same spatial region in another subband. Specifically, Shapiro noticed that in many cases, a low coefficient value in a low-frequency subband was correlated with low values for all coefficients relating to the same spatial location in higher-frequency subbands. This phenomenon is called a *zerotree* and commonly occurs in smooth regions of the image. Since each coefficient in a particular subband is related to four coefficients in the next-higher subband, recognizing and coding the occurence of a zerotree can considerably reduce the number of coefficients which need to be transmitted.

Formally, given a single coefficient (the tree *root*), we define a *tree* as the set containing the root and all coefficients in higher-frequency subbands corresponding to part of the spatial region described by the root. This is illustrated in Figure 3, in which a root is shown in subband HH_3 together with its child nodes in the high-frequency subbands. A zerotree is defined as a tree for which all nodes have absolute values lower than a given threshold.

The EZW algorithm is an embedded algorithm which encodes quantized wavelet coefficients in a bit-plane order (see Section 2.1). Each bit-plane is coded starting from the lowest-frequency subband, and on to the higher-frequency subbands. The encoder identifies zerotree roots and

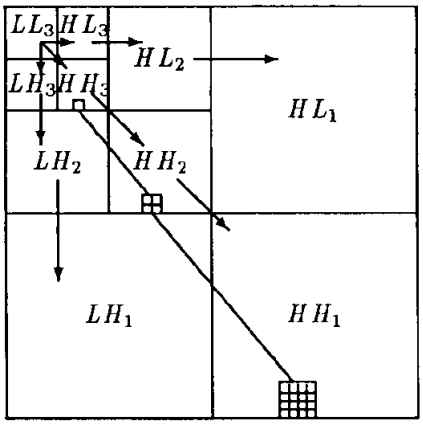

Figure 3: Example of a tree in the wavelet transform space (from [8])

indicates their position to the decoder using a special symbol, which indicates that other coefficients in the zerotree are not to be transmitted.

Specifically, the encoder functions as follows. The wavelet coefficients are divided into bit-planes, and processing begins with the most significant bit-plane. Within each bit-plane, bits are scanned in subbands, starting with the lowest-frequency subband. Bits are encoded differently depending on whether they are novelty bits or refinement bits (see Section 2), as follows. Refinement bits are encoded as is, i.e., using a '0' symbol and a '1' symbol. The encoding for novelty bits is more complex. If the bit belongs to the significance map (i.e., it satisfies the range condition (3)), then this information is transmitted to the decoder, along with the sign of the coefficient. Otherwise, the bit being examined is a '0'; this case is dealt with as follows. The bit is first checked to determine whether it is part of a zerotree. If it is a zerotree root, then this information is transmitted using a special "zerotree" symbol. If it is part of a zerotree, but not a zerotree root, then the decoder has already received a description of the zerotree, and therefore no symbol need be output for this bit. Finally, if the bit is zero but is not part of a zerotree, a "simple zero" symbol is transmitted.

The encoder output consists of symbols from two separate alphabets, used for encoding refinement and novelty bits, respectively. Since the decoder also keeps track of which bits are refinement bits, both sides are aware of which alphabet is being used for each coefficient. The refinement alphabet contains the symbols '0' and '1', while the novelty alphabet contains the following symbols:

- P: significant positive coefficient;

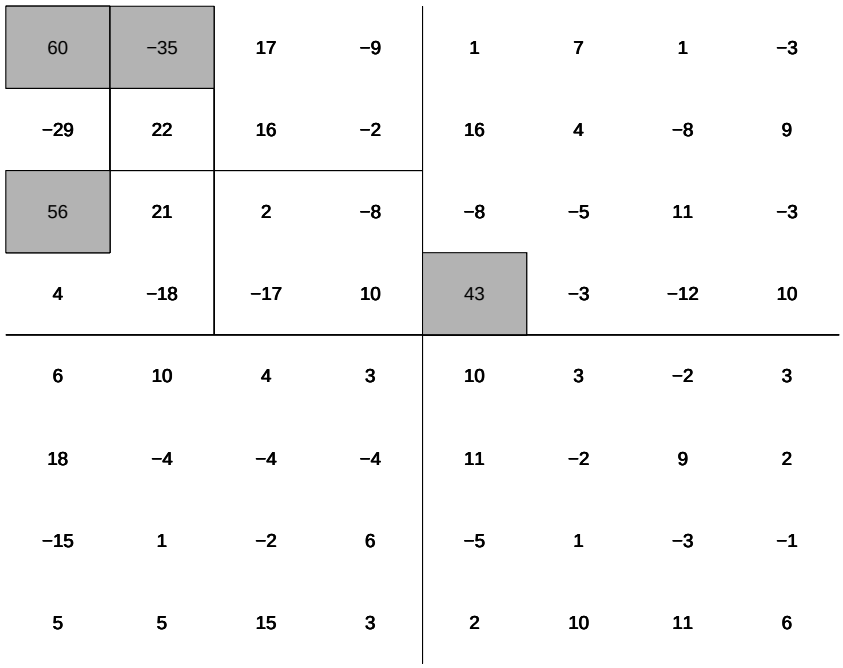

Figure 4: Numerical example of EZW encoding

- N: significant negative coefficient;
- Z: simple zero (an insignificant coefficient which is not part of a zerotree)
- T: zerotree root (an insignificant coefficient which is the root of a zerotree)

Both encoder and decoder must keep track of all zerotrees which have been identified to date. This allows the encoder to skip the transmission of bits which have already been identified as children in a zerotree, thus conserving bandwidth.

An additional performance improvement is achieved by marking all refinement bits as zero bits for the purpose of identifying zerotrees. Thus, if a particular large coefficient is surrounded by small coefficients, the (expensive) use of Z symbols to describe the surrounding occurs only for one bit-plane. After the large coefficient is identified as significant in a particular bit-plane, the surrounding region can be coded as a zerotree.

The algorithm is perhaps best illustrated with a numerical example. Consider the quantized wavelet coefficients in Figure 4. We describe the encoding for the most significant bit-plane, for which the threshold between 0-bits and 1-bits is 32. Significant bits are marked in gray in the figure. Most coefficients are described by 0-bits, and as we shall see, they can be encoded efficiently

using zerotrees. We begin with the lowest subband, which contains only the coefficient 60 (which exceeds the threshold), and is encoded as a P symbol. Subsequent bit-planes will encode this coefficient as a refinement coefficient. The next subband contains the coefficients -35, -29 and 22. Since the absolute value of -35 exceeds the threshold 32, it is encoded as an N symbol. The coefficient -29 does not exceed the threshold, and yet does not encode a zerotree (because of the child coefficient 56); it is thus encoded as a Z symbol. The coefficient 22, however, is the root of a zerotree: neither the root nor any of its child coefficients exceed the threshold. It is thus coded as a T symbol. In the coding of the subsequent subbands, the coefficients in this zero tree will not be coded. The encoding now proceeds in a similar fashion for the higher-frequency subbands. The complete encoding for the most significant bit-plane, assuming a raster-scan order, is as follows. A dot ($\cdot$) is used to indicate a coefficient for which no symbol is sent.

$$\underbrace{P}_{LL_0}\ \underbrace{N}_{LH_0}\ \underbrace{Z}_{HL_0}\ \underbrace{T}_{HH_0}\ \underbrace{TTZT}_{LH_1}\ \underbrace{PTTT}_{HL_1}\ \underbrace{\cdots\cdot}_{HH_1}\ \underbrace{\cdot\cdots\cdot\cdots\cdot ZZPZ\cdots}_{LH_2}\ \underbrace{ZZZZ\cdots\cdots\cdots\cdots}_{HL_2}\ \underbrace{\cdots\cdots\cdots\cdots\cdots\cdot}_{HH_3}$$

In summary, the EZW algorithm is an embedded algorithm for encoding wavelet transform coefficients. Its power lies in the use of zerotrees to efficiently encode large regions of insignificant coefficients.

3 Compression using Morphological Representation

Zerotrees are used as a method for efficiently representing large regions of insignificant coefficients in the wavelet transform. For example, Figure 5 (left) demonstrates the use of zerotrees to describe large regions of coefficients as insignificant using a single transmitted symbol. The zerotree concept serves as a basis for several enhancements to the EZW algorithm, such as the SPIHT method proposed by Said and Perlman [5]. However, as a description of rather amorphous shapes within the subband, it seems intuitively that zerotree methods may be lacking. This is because of the inherent structure induced by a zerotree: the basic building block of insignificant regions is a square, which is not the most efficient way to describe complex shapes; worse, the use of a diadic transform further restricts the squares to those having coordinates which are powers of two.

The problem at hand is therefore to describe the significance map (the set of coefficients exceeding a given threshold) more accurately, without increasing the bandwidth required. One such approach is suggested by Servetto et al. [6]. They propose to describe the significance map mor-

Figure 5: Zerotree vs. morphological description of the significance map (from [6])

phologically, i.e., in terms of generic geometric structure. If one can find an efficient technique for representing general shapes, then "mostly significant" and "mostly insignificant" regions can be identified within each subband (Figure 5, right). Since each of these regions would contain relatively homogeneous coefficients, entropy-coding each region separately could yield improved coding efficiency.

This claim is supported to some extent by several experiments performed on test images [6]. These experiments show that dividing the wavelet transform coefficients to significant and insignificant regions using morphological operations can provide improved coding gain. To some extent, these experiments are theoretical; for instance, they do not take into account the added bandwidth required for transmission of the shape of the coefficients being used. They do indicate, however, that a morphological approach to wavelet image compression is worth investigating.

The idea of morphologically describing amorphous regions of large or small wavelet coefficients is termed Morphological Representation of Wavelet Data (MRWD). We now present the general principles of MRWD; a detailed description of two algorithms implementing MRWD follows in the next subsections.

Morphological representation is useful when the image in question contains regions of significant coefficients. Its power lies in using the fact that when some coefficients are significant, neighboring coefficients are likely to be significant too. Thus, morphological representation arises from the

attempt to exploit intra-subband dependencies, in addition to the inter-subband dependencies which are the main focus of the zerotree approach. This dependency is implemented using the morphological idea of a *dilation operator*. The dilation operation simply means that if a coefficient is known to be significant, than all coefficients within a certain radius are also predicted to be significant. Predicted-significant coefficients are encoded using a separate probability table, which assigns higher probability (and hence less bits) for significant coefficients. Assuming that the prediction is generally correct, such a coding scheme will perform better than one which ignores intra-subband information.

Recall that much of the power of the EZW approach lies in the utilization of inter-subband dependencies, i.e., correlation between coefficients at different subbands which describe the same spatial location. MRWD also incorporates inter-subband prediction; this is done as follows. Subbands are encoded from lowest-frequency to highest-frequency, as in the EZW algorithm. Each encoded subband is used to predict the significance of the next-higher subband: a significant coefficient indicates a higher probability for significant coefficients in the same spatial location, for the next-higher subband. As in the intra-subband dependencies, predicted significant coefficients are encoded using a separate probability table, which assigns less bits for a significant coefficient.

Based on the idea of MRWD, two image compression algorithms have been proposed [6]. The *single-rate* algorithm is a quality-based algorithm which is designed to achieve a certain MSE level, regardless of resulting bitrate. The second is an *embedded* algorithm which can be used to achieve any desired bitrate. Both are based on the morphological ideas presented above, but the implementation details differ. A complete description of each algorithm follows. It should be noted that some details of the implementation were not described in the paper itself, and these were (painstakingly) reconstructed from the C source code accompanying the paper. Further details regarding the inaccuracies in the paper are given in Section 4.1.

3.1 Single-Rate Algorithm

The purpose of the single-rate algorithm is to provide a simple image compression algorithm based on morphological principles, with a prestated accuracy goal (as opposed to an embedded approach). The reconstruction accuracy is controlled using quantization of the transform coefficients; after quantization, these coefficients are transmitted losslessly to the decoder. The morphological aspects

of the encoding algorithm refer only to the method of predicting coefficient values, which reduces the number of bits required for transmission.

The quantizer used is known as a uniform scalar quantizer with a deadzone. This quantizer encodes each coefficient independently of other coefficients (*scalar* quantization). A *uniform* step size q, chosen by the user, is used; however, the width T of the region to be quantized to zero (called the *deadzone*) is larger than a standard quantization bin. The deadzone approach [9, 5] is known to improve the rate/distortion of the image by descreasing entropy without significantly reducing subjective image quality. Empirical tests [6] have found that $T = 1.4q$ yields optimal performance.

The effect of the quantizer on a coefficient c can be summarized as

$$c_q = \begin{cases} 0 & |c| < T/2 \\ [c/q] & |c| \geq T/2, \end{cases} \tag{4}$$

where $[x]$ indicates rounding x to the nearest integer.

After quantization, the set of coefficients is transmitted losslessly to the decoder. For this purpose, "significant" and "insignificant" coefficients are distinguished by magnitude. Coefficients which fall into the three centermost bins (i.e., coefficients in the range $|c| < 1.5q$) are termed insignificant, while all other coefficients are termed significant. Morphological properties are used in an attempt to predict the significance of coefficients. This process is described in detail in the remainder of this subsection.

We first list the symbol alphabets used in the encoding process. MRWD uses different symbol alphabets for encoding significant and insignificant coefficients, as follows:

- *Complete Alphabet:* This alphabet can encode the full range of coefficients in the transform. It is used for encoding significant coefficients, and during morphological scans. This alphabet is referred to as the `t`-table in the MRWD source code.

- *Significance Verification Alphabet:* This alphabet can encode the three insignificant symbols -1, 0, and 1, as well as a special symbol S used to indicate that a coefficient is significant. It is used for predicted-significant symbols. The alphabet is referred to as the `t2`-table in the MRWD source code.

- *Insignificance Verification Alphabet:* This alphabet can encode only the symbols 0 and S. It is used for predicted-insignificant symbols, and its used is described in detail below. It is

referred to as the `tt2`-table in the MRWD source code.

Having defined the alphabets used in the encoding process, we can now describe the encoding algorithm. The algorithm begins by encoding the lowest-frequency subband in a simple raster scan order, using the complete alphabet. This serves as a basis for predicting significance in higher subbands. Encoding now proceeds for all other subbands, in order of increasing frequency. The following steps are performed for each subband:

1. A table storing the set of transmitted coefficients in the subband is initialized to the empty set.

2. A prediction mask is computed based on results from the parent subband, i.e., the next-lower frequency subband having the same frequency direction. A coefficient is predicted to be significant if its matching coefficient in the parent subband is either itself significant, or lies close to a significant coefficient. Closeness is defined in an 8-nearest-neighbors sense.

   Thus the obtained prediction mask contains not only those coefficients which were significant in the parent subband, but also neighboring coefficients. This can be viewed as performing a trivial expansion of the significance mask from the parent to the child subband, and then applying a morphological dilation operation [3] on the resulting prediction mask.

3. The adaptive probability tables for all alphabets are reset.

4. Predicted significant coefficients in the subband are scanned in a raster order. Each coefficient is checked to determine whether it has already been transmitted. If it has not, an appropriate symbol from the Significance Verification Alphabet is transmitted: the exact value in case of an insignificant coefficient, and the S symbol in case of a significant coefficient. If a significant coefficient is found, its exact value is then transmitted using the Complete Alphabet, and morphological scanning of neighboring coefficients begins (see below).

5. Before a scan of predicted insignificant coefficients begins, the adaptive probability tables for all alphabets are reset once again.

6. Predicted insignificant coefficients in the subband are now scanned in a raster order. Each coefficient is checked to determine whether it has already been transmitted. If it has not,

an appropriate symbol from the Insignificance Verification Alphabet is transmitted: 0 for insignificant coefficients or S for significant coefficients. The insignificant values -1 and 1 are also encoded as 0 in this stage. If a significant coefficient is found, its exact value is then transmitted using the Complete Alphabet, and morphological scanning of neighboring coefficients begins (see below).

The heart of the encoding algorithm lies in the *morphological scan* which occurs whenever a significant coefficient is identified. As we have seen, once a significant coefficient is identified, chances are that surrounding coefficients will also be significant. Thus, the encoding algorithm stops the raster scan of coefficients, and begins a morphological scan in which the eight nearest neighbors of the significant coefficient are scanned. The encoder checks whether any of these coefficients has not been transmitted yet. If such a coefficient is found, its value is encoded using the Complete Alphabet, and morphological scanning begins recursively from the newly-identified significant coefficient. This approach allows the encoder to describe amorphous (but connected) regions using a morphological growth operation.

As an example, suppose we are to perform a morphological scan on the coefficients given in Figure 6(a), starting with the coefficient 40. Suppose that no coefficient has been transmitted yet. After transmitting the value 40, the encoder scans the surrounding coefficients, and since none of them has been transmitted yet, all eight neighbors are transmitted in the first stage of the morphological scan (Figure 6(b)). The transmitted symbols are listed in the bottom of Figure 6(b). These coefficients are then marked as transmitted (indicated by a black dot in Figure 6(c)), and the morphological scan is recursively applied to each of the newly identified significant coefficients. This begins with the coefficient 25, leading to the transmission of its neighboring coefficients (except for those which have already been transmitted), as shown in Figure 6(c). Among these coefficients, only the value -5 is significant, and its neighboring coefficients are transmitted as well (Figure 6(d)). Since none of its neighbors are significant, encoding returns to higher levels of the recursion, in the case the coefficient -20 (Figure 6(e)). Encoding continues in a similar manner until no more significant coefficients remain (Figure 6(f)–(g)).

A curious point in this algorithm lies in the choice of a different alphabet for predicted significant and predicted insignificant coefficients. While it makes sense to use different *probability tables* for the two cases, the algorithm introduces a further difference: the Predicted Insignificant Table does not

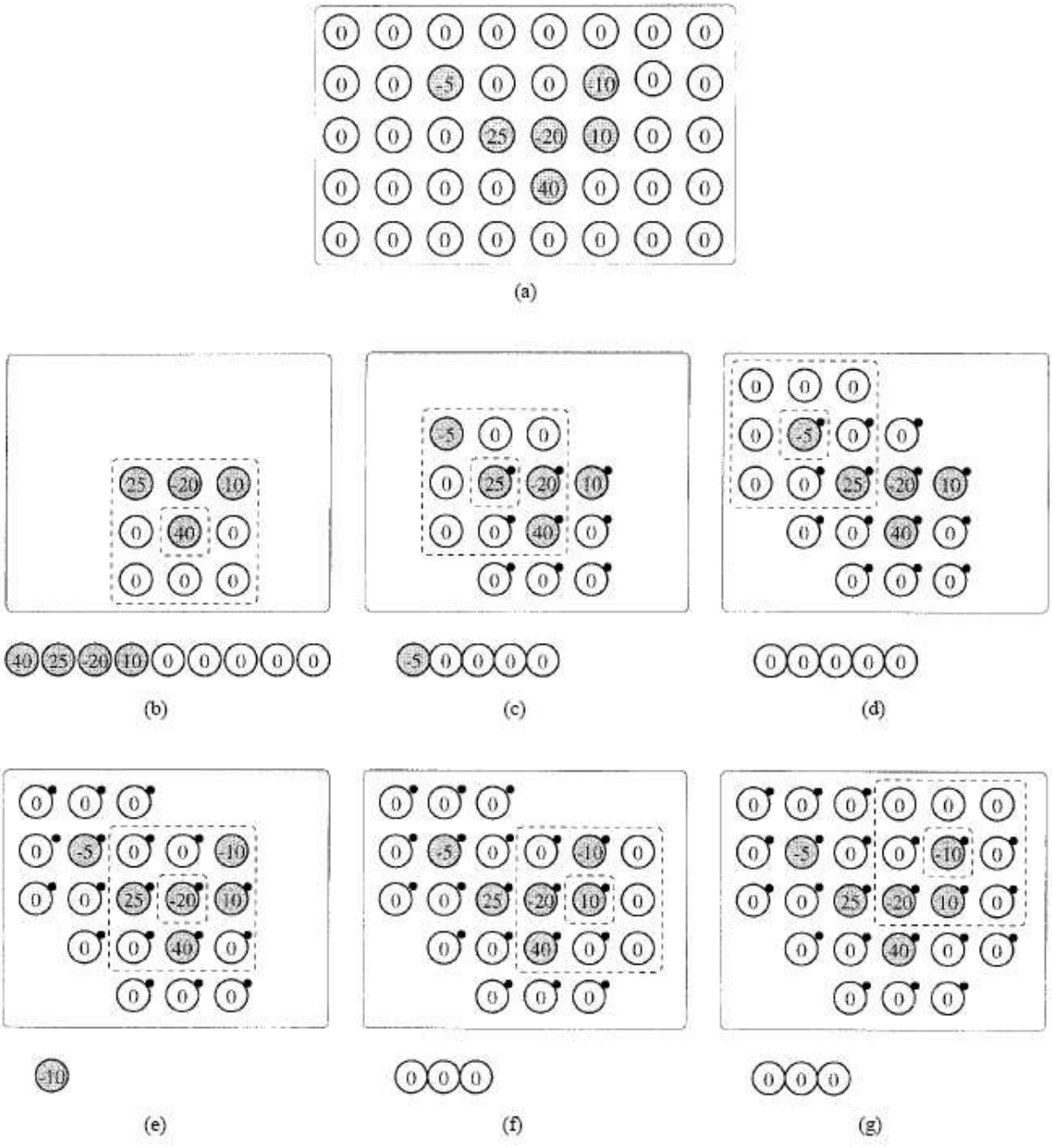

Figure 6: Numerical example of the morphological scan (from [6])

contain symbols for coefficient values -1 and 1. These values are coded as 0 values. The authors of the paper explain this choice briefly as arising from "entropy constraining" considerations, i.e., this is an attempt to reduce the entropy of coded symbols without significantly reducing image quality. Nevertheless, this choice seems questionable. The entropy-constraining algorithm is applied only to predicted-insignificant coefficients. It seems unreasonable to expect insignificant coefficients to behave differently depending on whether they were predicted insignificant or not. Applying this choice to insignificant coefficients, regardless of their prediction, results in the mechanism of the deadzone, which has already been put to use in the quantizer. Thus, the value of this added entropy-constraining algorithm seems arguable at best, and Servetto et al. report no attempt to test its effectiveness, though such a test is certainly in place.

3.2 Embedded Algorithm

Servetto et al. [6] also suggest an embedded MRWD algorithm. As in the EZW algorithm, embedding is achieved by encoding bit-planes in sequence, and encoding is performed separately for novelty and refinement bits (see Section 2.1). However, the mechanism for encoding novelty bits is based on morphology, rather than on zerotrees. While the morphological principles are similar to those of the single-rate algorithm, some differences remain; this algorithm is therefore described independently.

We first describe the quantization strategy. The quantization describes each coefficient in a sequence of bit-planes, each having a different coefficient $T_n = 2^n$. The quantization of a coefficient c in a bit-plane n is performed as follows:

- If $|c| < T_n$, the coefficient is quantized as I, indicating an insignificant value.
- If $|c| \geq T_n$, and all higher bit-planes quantized the coefficient as I, then the coefficient is quantized as $+$ or $-$, depending on its sign.
- Otherwise, the next most significant bit is quantized using the symbols $+$ and $-$.

The first two items in this quantization algorithm correspond to novelty bits, which are quantized using the symbols $\{I, +, -\}$. The last item corresponds to refinement bits, which are quantized using the symbols $\{+, -\}$.

As in the single-rate algorithm, several alphabets are used to transmit these symbols; the choice of an alphabet depends on the context. However, different algorithms are used in the embedded case. These alphabets are:

- *Significant Alphabet:* This alphabet is used for encoding refinement bits and bits which are known to be significant. It contains the symbols $\{+, -\}$.
- *Growth Alphabet:* This alphabet is used to indicate bit significance for the morphological growth routine. It contains the symbols $\{I, \pm\}$, with $\pm$ indicating that the bit is either of type $+$ or of type $-$.
- *Prediction Alphabet:* This alphabet is identical to the growth alphabet, but is used for indicating significance based on the prediction mask.

Notice the use of separate alphabets for indicating significance with the growth and prediction routines. This separation is used in case the two routines have different probabilities of success.

The encoding process is performed separately for each bit-plane, starting with the most significant bit-plane. For each bit-plane, coefficients are scanned in subbands, in order of increasing frequency. The coefficients in the lowest-frequency subband defined as refinement bits from the very first bit-plane, and they are transmitted using the Significant Alphabet. For each subband other than the lowest, the following operations are performed:

1. The probability tables of all alphabets are reset. Refinement bits are transmitted using the Significant Algorithm.
2. The probability tables of all alphabets are reset. A morphological growth routine (see below) is applied to each refinement bit.
3. A prediction mask is generated for the subband, with the same method used for the single-rate algorithm.
4. The probability tables of all alphabets are reset. Each predicted coefficient is examined to see whether it has already been transmitted. If not, its significance is transmitted using the Prediction Alphabet. If the coefficient is significant, its value is transmitted using the Significant Alphabet, and a morphological growth routine (see below) is applied.

5. The probability tables of all alphabets are reset. All coefficients which have not been transmitted yet are scanned. The significance of each of these coefficients is transmitted using the Prediction Alphabet. If the coefficient is significant, its value is transmitted using the Significant Alphabet, and a morphological growth routine (see below) is applied.

The morphological growth routine in the algorithm above functions as follows. Given a starting position, all neighboring coefficients are examined to see whether they have already been transmitted. The significance of those which have not been transmitted is transmitted using the Growth Alphabet. Any coefficient which is found to be significant is encoded using the Significant Alphabet, and a morphological growth routine is recursively applied to it. As in the single-rate algorithm, this approach yields an efficient encoding of amorphous connected regions in the subband.

4 Analysis

This section describes the author's impression of the paper under review, of the idea of morphological representation of wavelet data, and of the current status of research on wavelet image compression. We begin with a discussion regarding the clarity and conciseness of the paper, and continue with aspects relating to the contents of the paper.

4.1 Organization of the Paper

The paper is organized into three main sections. First, the motivation for morphological representation is presented. This includes background in existing wavelet compression techniques, and some simple experiments showing the potential gains to be achieved using the morphological approach. Second, two suggestions for MRWD algorithms are presented. Finally, simulations are performed to compare the proposed algorithms to existing methods.

The first section, containing the introduction and motivation, is presented clearly and convincingly. Current methods are reviewed fairly, and experiments demonstrate the possibility that morphological representation of wavelet data could achieve substantial coding gain (although, as admitted by the authors, these experiments are somewhat theoretical). The third section, comparing MRWD with other compression methods, is also well-presented; when comparing MRWD to the classical zerotree approach, the authors honestly admit that "no inference can be made regarding

the superiority of one method over the other" [6, p.1173].

Compared with these two well-written parts of the paper, the middle section, describing the MRWD algorithms themselves, seems somewhat obscure. The inner workings are the algorithm are described in inaccurate and sometimes contradictory terms. Several examples of these problems are presented below.

One of the aspects of the algorithm which is very difficult to understand is the use of numerous symbol alphabets in the encoding process. In no place is there a clear description of the different alphabets used, when each alphabet is to be used and why different alphabets are used in similar contexts. For instance, there is no reference to the fact that the Significance Verification Alphabet and the Insignificance Verification Alphabet (Section 3.1) perform nearly identical purposes, yet have a different number of symbols in them. These alphabets are not even mentioned by name in the paper; only an implicit reference is given to their existence.

Some parts of the algorithm are described in more detail, but these sometimes suffer from inaccurate wording. Consider the following example, describing the idea of the morphological growth operation:

> "[T]he encoder scans subbands in a raster scan order, until a significant coefficient is detected. In such a situation, the encoder signals this event to the decoder with a special symbol.... With both encoder and decoder aware that a significant coefficient is present at the current location..., the encoder sends to the decoder and labels significant those coefficients in a neighborhood of the current significant one. Once encoder and decoder have access to a few extra coefficients, these new ones are examined: if new significant coefficients are found, the process is applied recursively...." [6, p.1167]

This paragraph is unclear because the word "significant" is used with two different meanings: first, it is used to indicate that coefficients are *predicted* to be significant if they neighbor a significant coefficient. Then it is used to indicate *correctly predicted* significant coefficients. Without this distinction, the above paragraph can only be understood to be saying, "The significant coefficients are scanned to determine if they are significant." The distinction between predicted significant coefficients and actually significant coefficients is unclear in other places in the paper as well.

An engineer attempting to implement the proposed compression algorithm in this paper would thus have a hard time understanding the algorithm from the text. The algorithm can sometimes

be more succintly expressed as pseudocode, which is indeed supplied in the paper. Unfortunately, pseudocode is only given for part of the algorithm (the morphological growth routine). In addition, the pseudocode contains symbols whose meaning is not described at any point in the paper (such as the parameter `ACContext`). A more complete pseudocode listing could provide better understanding of the algorithm.

In summary, reading Servetto et al.'s paper, one can clearly understand both the ideas leading to MRWD-based algorithms, and the fair results obtained using the algorithm. However, much remains unclear as to specifics of the operation of the algorithm. The writer of this report had to download MRWD source code, supplied by the authors of the paper, and examine it in detail, in order to fully understand the algorithm. Clearly this should not be the aim of the paper's authors.

4.2 Comparison with Classical Algorithms

The single-rate and embedded MRWD compression algorithms were tested on a set of images, in order to compare their performance to the classical EZW compression algorithm.

Procedure. Source code for the MRWD algorithms was obtained from S.D. Servetto's web site [7] and was modified for compilation on a PC platform. Source code for the EZW algorithm was obtained from a N. Mow-Song's web site [4].

Five images, photographed by the author and his family, were used for the testing procedure. Each image was converted to grayscale and trimmed to a standard size of 512×512 pixels. The images, labeled Carmel, Cactus, Forest, Puzzle, and Tulip, were chosen to represent different applications:

- Forest, Carmel and Cactus are images with many details and sharp features, and are relatively difficult to compress;
- Tulip is a relatively smooth image;
- Puzzle consists of several disjoint objects (puzzle pieces) over a smooth object, intended to simulate a medical image.

In all algorithms, the 5-level Daubechies 7/9 wavelet transform was used, as recommended by [6]. Four different bit rates were tested between 1.0 and 0.1 bits per pixel. In the embedded

algorithms, this was performed by deleting all bits of the image appearing later than the allowed file size. In the single-rate algorithm, the compression factor was modified iteratively until the correct file size was obtained.

Results were analyzed using both objective and subjective criteria. The objective criterion used was the peak SNR (PSNR), defined as

$$\mathrm{PSNR} = \frac{255^2}{\mathrm{MSE}} = \frac{255^2}{\frac{1}{N}\sum_i \sum_j (p_{ij} - \hat{p}_{ij})^2}, \tag{5}$$

where $\mathbf{p}$ is the image to be compressed, $\hat{\mathbf{p}}$ is its reconstruction, and 255 is the maximum (peak) value of the signal.

Reconstructed images were also examined subjectively to determine the types of artifacts formed by the various compression algorithms, and to find the compression algorithm which results in optimal subjective quality.

Results. Because of the large number of images obtained when running the simulation, and because of limitations of print quality, it is not feasible to include all images performed in the simulation. Instead, the reader is referred to the author's web site `http://www.technion.ac.il/~zvikabh/wavelet`, which contains all results. A single example is given in Figure 7 to illustrate some subjective properties.

Objective Analysis. PSNR values for the various images, compression algorithms and bit rates are summarized in Figure 8. As can be seen from these results, the primary difference in image quality results in differences between images: some images (such as CACTUS) are more difficult to compress than other images (such as TULIP). This is an expected result of the fact that some images contain many details, while others are relatively smooth.

For each image, PSNR can be compared among compression algorithms. The PSNR difference between the MRWD algorithms and the EZW algorithm is summarized in Table 1. The table lists the mean and standard deviation of the PSNR increase, in dB.

The comparison reveals that the MRWD algorithms perform significantly better than the EZW algorithm, with little variation between the two MRWD algorithms themselves. The use of MRWD results in a typical increase of 0.5–1.0 dB PSNR, compared with EZW at the same rate. Furthermore, the single-rate and embedded MRWD algorithms provide nearly identical PSNR, with

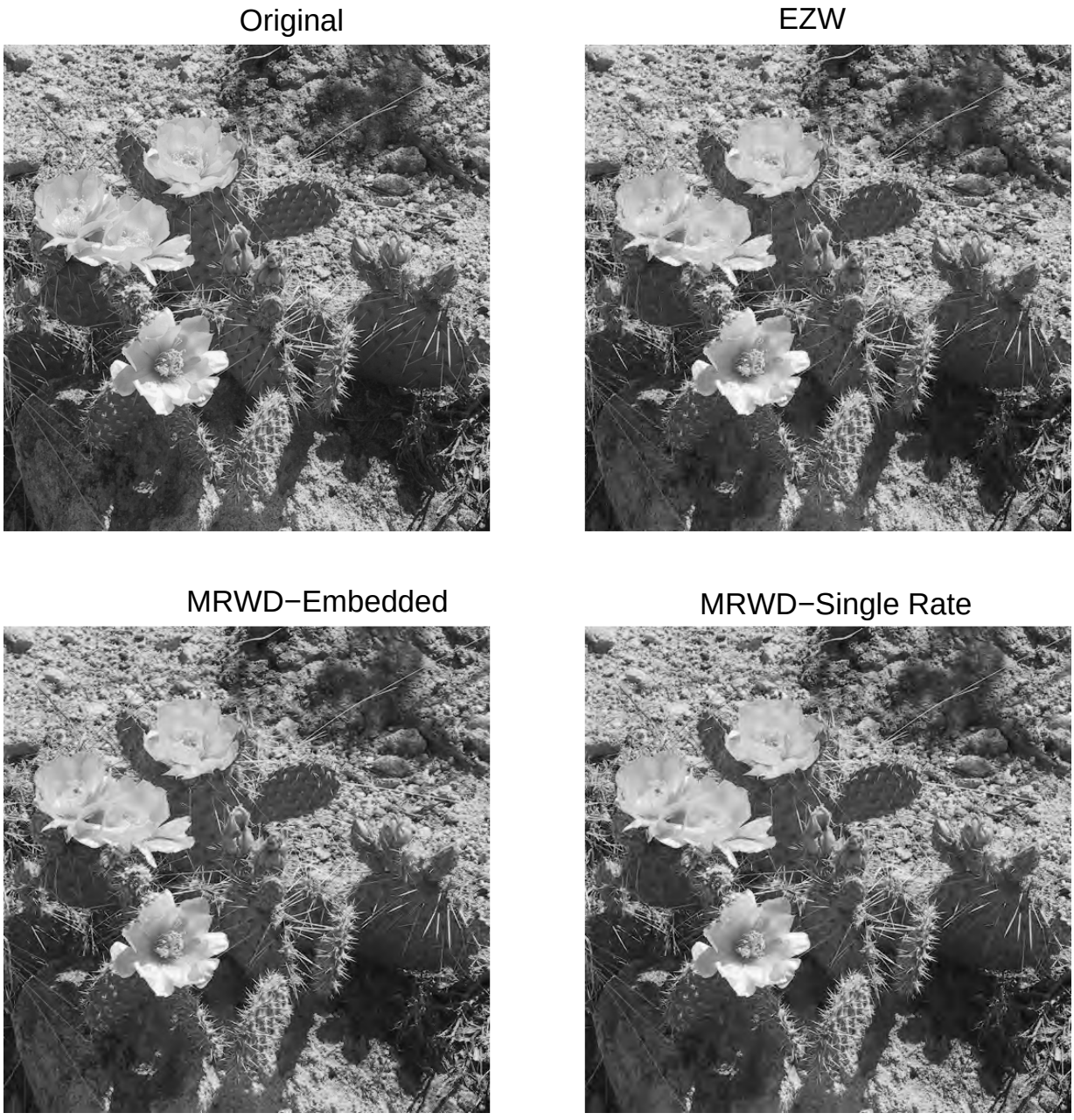

Figure 7: Cactus at 1 bit per pixel, with various compression algorithms

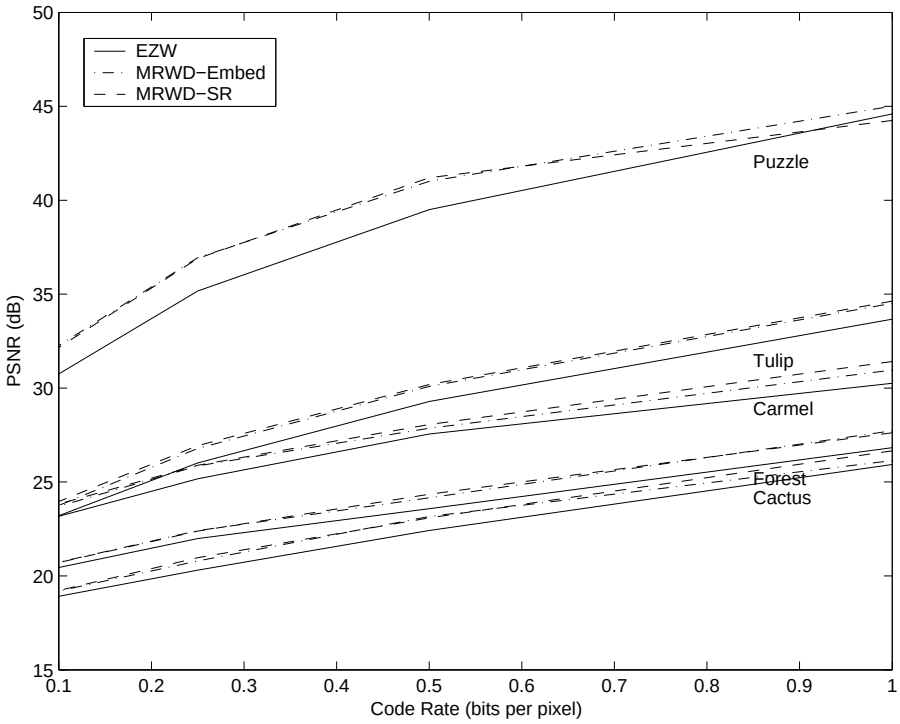

Figure 8: Comparison of PSNR values for different compression algorithms

Algorithm	Code Rate, in bits per pixel			
	1.00	0.50	0.25	0.10
MRWD Single Rate	0.60 (0.29)	0.78 (0.44)	0.84 (0.56)	0.64 (0.50)
MRWD Embedded	0.68 (0.58)	0.92 (0.46)	0.88 (0.49)	0.70 (0.45)

Table 1: PSNR Increase over EZW for Different Code Rates, in dB

(Numbers in parentheses indicate the standard deviation)

a slight advantage for the embedded algorithm. The advantage of the embedded algorithm, which was not mentioned by the authors of the MRWD paper, is more pronounced in the subjective analysis, and will be dicussed further presently.

Subjective Analysis. The reconstructed images were examined by eye to determine the subjective quality of each algorithm. For each image, the three algorithms under consideration were ranked for subjective image quality. To avoid bias toward a particular algorithm, a blindfold approach was used: the evaluator did not know which compression algorithm was used for each image, until evaluation was complete. Reconstructed images were compared based on two criteria:

- **Sharpness:** Image is well-focused. Small details and texture are sharp and clear. Contours and edges are accurate and well-defined.
- **Fidelity:** Image does not contain distracting artifacts. Outstanding features of the original image are not obscured by sharp reconstruction errors.

The first criterion requires an algorithm which recreates sharp edges, while the second is generally achieved by smoothing the resulting image. Thus, it is not surprising that these two criteria were often contradictory. In these cases, algorithms were chosen based on the overall subjective quality of the image. Clearly, this grading method is less accurate than a scientific comparison such as PSNR; a more complete study should probably perform subjective tests on multiple subjects and use statistical tools to estimate the reliability of the results. Still, as we shall see, some insight can be gained from this simple experiment, insight which may not be evident from the objective tests performed.

The results are presented in Table 2. In this table, the three algorithms are ranked from best to worst for each image and bit rate.

Despite the innacurate nature of subjective comparisons, some significant phenomena can be observed from these results. A clear distinction between the complex images (Cactus, Carmel and Forest) and the simpler images (Puzzle and Tulip) is evident. In all tests, the complex images were reconstructed better with the EZW algorithm than with either of the MRWD algorithms. Contrariwise, the simple images were more adequately reconstructed with the morphological algorithms.

Image	Code Rate, in bits per pixel			
	1.00	0.50	0.25	0.10
Cactus	EZW EMB SR	EZW SR EMB	EMB EZW SR	N/A†
Carmel	EZW EMB SR	EZW EMB SR	EMB SR EZW	N/A†
Forest	EMB EZW SR	EZW EMB SR	EMB EZW SR	N/A†
Puzzle	N/A‡	EMB SR EZW	SR EMB EZW	EMB SR EZW
Tulip	N/A‡	SR EMB EZW	SR EMB EZW	EMB SR EZW

Table 2: Subjective comparison of compression algorithms, from best to worst
(EZW = Embedded Zerotree Wavelet Compression; EMB = Embedded MRWD Compression;
SR = Single-Rate MRWD Compression)

† Image quality was too low for a quality comparison.

‡ Image quality was too high for a quality comparison: images appear identical.

This difference can be explained from the fact that complex images are not clearly divided into significant and insignificant shapes. The use of the morphological dilation operator in these cases may result in a detrimental compression quality, because of failure to predict significant coefficients. However, simpler images can successfully be divided into morphological regions, and these can be efficiently described using the MRWD algorithm. This effect is most apparent in the PUZZLE image, in which artifacts resulting from the use of square-shaped zerotrees are clearly visible at low bit rates with the EZW algorithm, while MRWD maintains image quality.

In summary, despite the fact that MRWD outperformed EZW in terms of PSNR, its subjective quality was, in many cases, lower than that of the the standard EZW algorithm. This raises questions not only regarding the quality of the MRWD algorithm, but also regarding the very common use of PSNR as a measure of image quality.

By subjectively examining the images it is also clear that the MRWD algorithms tend to generate smoother reconstructed images. This may be advantageous at low bit rates, in which sharp features in the reconstructed image are often artifacts. However, for complex images at moderate to high bit rates, the smoothing effect generally reduces the ability to discern image details (see, for example, the cactus thorns in Figure 7) and damages textures (as in the tree trunk in FOREST).

Another surprising find is that the embedded MRWD algorithm functioned at least as well as the single-rate algorithm. This was apparently unknown to [6], who wrote that "[The single-rate algorithm] is built to achieve high performance at a single target bit rate" [6, p. 1166], implying some sort of optimization not performed in the embedded algorithm. It appears that the single-rate algorithm is no more successful than the embedded algorithm even for non-embedded applications. This may be a result of the fact that the single-rate algorithm uses much larger alphabets for the arithmetic coding process (some of the alphabets contain all possible quantized values). As recognized by [8], the problem with such alphabets is that estimation of symbol probabilities is more complex, and thus convergence to entropy-level coding is slower. Since the codebook is reset several times per subband, it is possible that the arithmetic coder is reset before the probability tables converge, resulting in low compression levels.

5 Summary and Conclusions

We conclude with a summary of the tests performed on the compression algorithms, some suggestions for improvements to the MRWD algorithm, and some thoughts on the use of objective quality measures such as the PSNR.

5.1 Algorithm Comparison

The morphological compression algorithm proposed by Servetto et al. [6] is a compression algorithm based on identifying large-coefficient regions or shapes in each wavelet transform subband, and efficiently coding these shapes using several prediction techniques. This approach was compared with the classical embedded zerotree wavelet (EZW) compression algorithm [8]. It was found that the PSNR of morphology-compressed images was higher by 0.5–1.0 dB than that of EZW-compressed images. This improvement was evident in different images and various bit rates, and is a strong point in favor of MRWD algorithms.

Nevertheless, an in-depth manual inspection of images compressed using the two algorithms reveals that in some cases, the EZW algorithm may outperform morphological algorithms in terms of subjective image quality. MRWD is well-suited for images with large, smooth regions, which can be efficiently identified and coded using morphological techniques. However, more complex images cannot always be divided into large significant shapes, and the morphological algorithms sometimes fail to encode such images efficiently. While this is apparently not a significant cause of error in terms of PSNR, the subjective image quality is greatly reduced.

It is therefore not possible to reach a catch-all conclusion regarding the use of MRWD and EZW. Either may be preferable, depending on image properties.

A comparison between the single-rate and embedded MRWD algorithms was also performed. In terms of PSNR, these algorithms are nearly identical; yet the embedded algorithm almost always results in more detailed images, while the single-rate algorithm yields smoother images. In some images a smoothing effect can improve subjective image quality, but most subjective tests showed a significant preference for the embedded algorithm. A possible explanation is that embedded encoding may be more efficient than single-rate encoding because of the use of smaller symbol alphabets. This is a point which deserves further study, as an understanding of coding inefficiencies

may improve future compression techniques.

5.2 Alternative Morphological Algorithms

While this report did not find conclusive results in favor of morphological compression algorithms, the results certainly justify continuing research effort. Some suggestions which may improve the performance of morphological algorithms follow.

Anisotropic Dilation Operators. The MRWD algorithms use isotropic dilation operators for predicting coefficient significance. In other words, if a coefficient is found to be significant, then all neighboring coefficients are predicted to be significant as well. However, horizontal frequency subbands tend to have more horizontal significance shapes, while vertical frequency subbands tend to have more vertically-shaped significance regions. This can be seen, for example, in the wavelet transform in Figure 2 (left). This fact can be exploited using anisotropic dilation operators, for example, a dilation operator with four neighbors in the preferred direction of the subband and two neighbors in the perpendicular direction. Such dilation operators may increase the probability of correctly predicting significant coefficients.

Alternative Shape Descriptors. Another topic for further research is the method of representing significant regions in the encoded data stream. It is possible that alternative methods for shape representation could perform quite well for describing shapes in the wavelet transform. For instance, the authors mention chain codes as a possible shape descriptor, but dismiss this idea as being "very expensive in bit rate" [6, p.1167]. Chain codes are shape descriptors consisting of a chain of directions, such as LEFT, DOWN, etc. They are used to define the outline of a shape, after which its interior is transmitted using standard coding techniques. Although they require additional side information, they have the advantage that anything left outside of the defined shape is known to have a value of zero, and thus need not be transmitted; at the same time, all coefficients inside the defined shape are known to be significant, thus eliminating the problem of incorrectly predicted coefficients. Whether or not such an approach could perform as well or better than the morphological approach remains a question for further study.

5.3 Image Quality Measures

As we have seen, much additional insight can be gained from subjective analysis of compressed images. Important conclusions, which were not evident when comparing PSNR values, arise when observing images by eye. This occured both when comparing morphological algorithms to the zerotree algorithm, and when comparing the embedded algorithm to the single-rate algorithm. It seems, then, that the PSNR may not be an optimal measure of image quality. It is well-known that the PSNR does not always indicate subjective quality, and yet this measure is used very often to measure image quality [5, 8, 6]. This may be due to the credibility given to a repeatable, objective test. While repeatability is certainly a desireable quality, the actual goal of most compression applications is subjective observer quality.

One possible explanation for the inadequacy of the PSNR measure is its sensitivity to the average energy in a particular region of the image. In a dark region of the image, the variance of the image is low, and thus the error will necessarily be low as well. Hence, attempting to maximize the PSNR will result in appropriating a large number of bits to light regions, with much less attention given to shadows and dark regions. This may not be an optimal strategy in terms of subjective quality, as can be seen in Figure 7, in which the most noticeable errors occur in shadows. If an image contains large dark regions, it is worth encoding some information from those regions too.

Wavelet image compression is based on the attempt to minimize the MSE, which is equivalent to maximizing the PSNR; this is the reason wavelet coefficients are uniformly encoded (Section 2.1). If a better measure of image quality can be found, wavelet compression algorithms might be able to improve the obtained subjective quality. In some research areas, it is common to use standardized human trials as an authoritative measure of quality. An example is the Mean Opinion Score (MOS) used for audio codec evaluation.

While a subjective score could function as a final test for a compression system, algorithm design requires more accurate knowledge of the objective. This, too, may be possible to achieve. For instance, the well-known JPEG compression algorithm [11] specifically quantizes high-frequency components more strongly than low-frequency components. Since the DCT transform used in JPEG is unitary, having a large error in some components and a small error in others does not make sense, from a MSE point of view. Yet this approach is used with the explicit assumption that the human visual system is less sensitive to higher frequencies. The compression is justified

based on physiological arguments, rather than an attempt to minimize a mathematical formula. It is worth examining whether a similar approach may be used for wavelet compression as well.

References

[1] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Proc.,* 1(2): 205–220, April 1992.

[2] R.A. Devore, B. Jawerth, and B.J. Lucier, "Image compression through wavelet transform coding," *IEEE Trans. Inf. Theory,* 38(2): 719–746, March 1992.

[3] P. Maragos and R. Schafer, "Morphological systems for multidimensional signal processing," *Proc. IEEE,* 78(4): 690–710, Apr. 1990.

[4] Mow-Song, N., EZW compression routines, `http://pesona.mmu.edu.my/~msng/EZW.html`

[5] A. Said and W.A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circ. Syst. Video Tech.,* 6(3): 243–250, June 1996.

[6] S.D. Servetto, K. Ramchandran, and M.T. Orchard, "Image coding based on a morphological representation of wavelet data," *IEEE Trans. Image Proc.,* 8(9): 1161–1174, Sep. 1999.

[7] S.D. Servetto, K. Ramchandran, and M.T. Orchard, MRWD compression routines, `http://cn.ece.cornell.edu/publications/papers/19961203/`

[8] J.M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Sig. Proc.,* 41(12): 3445–3462, Dec. 1993.

[9] D.S. Taubman and M.W. Marcellin, *JPEG 2000: Image Compression Fundamentals, Standards and Practice,* Kluwer International Series in Engineering and Computer Science, Secs 642.

[10] C. Valens, "EZW encoding," Online article, `http://perso.wanadoo.fr/polyvalens/clemens/ezw/ezw.html`

[11] G.K. Wallace, "The JPEG still-image compression standard," *Comm. ACM,* 34(4): 30–44, April 1991.

[12] J.Z. Wang, "Wavelets and imaging informatics: a review of the literature," Online article, `http://wang.ist.psu.edu`

[13] I.H. Witten, R. Neal and J.G. Cleary, "Arithmetic coding for data compression," *Comm. ACM,* 30(6): 520–540, June 1987.

[14] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Theory,* 23(3): 337–343, May 1977.